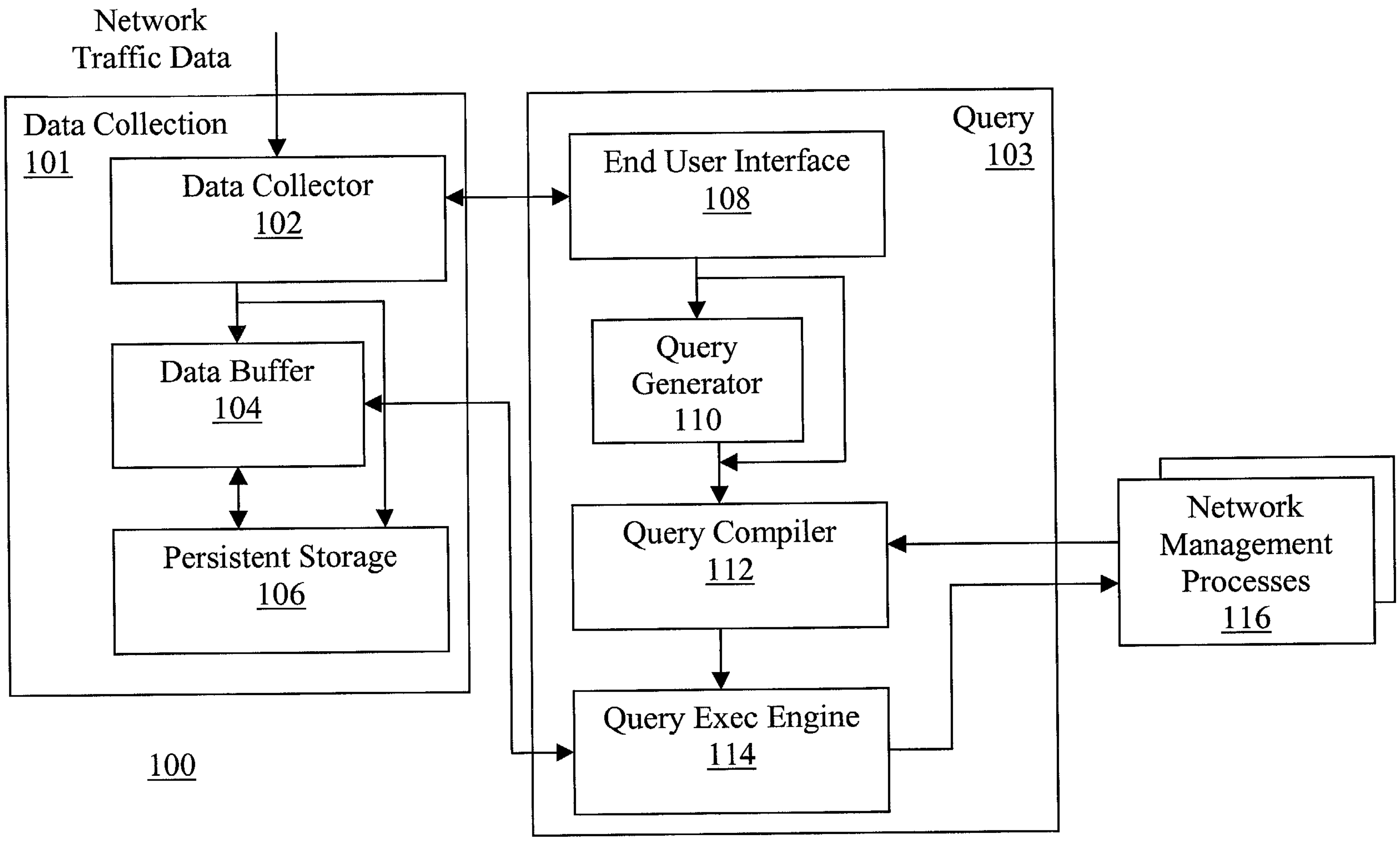

Network
Traffic Data
Data Collection
101
Data Collector
102
Data Buffer
104
Persistent Storage
106
100
Query
103
End User Interface
108
Query
Generator
110
Query Compiler
112
Query Exec Engine
114
Network
Management
Processes
116

Figure 1

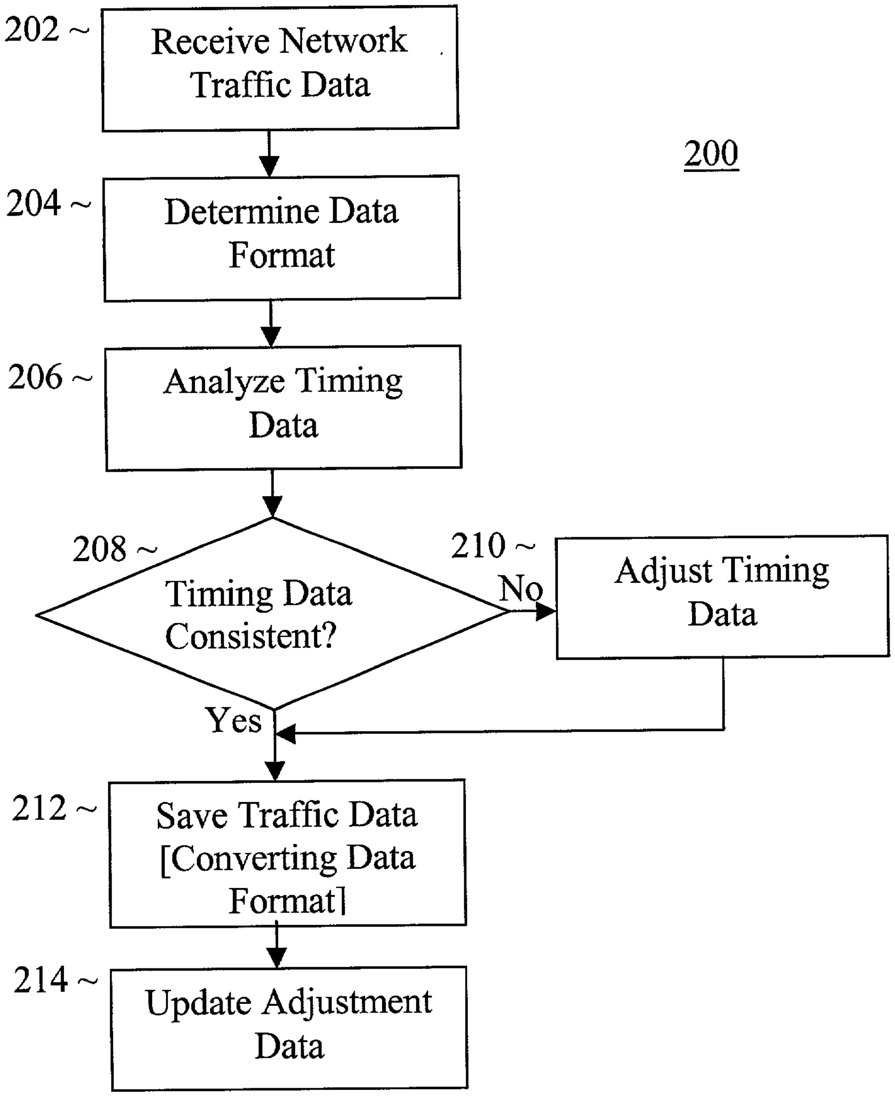
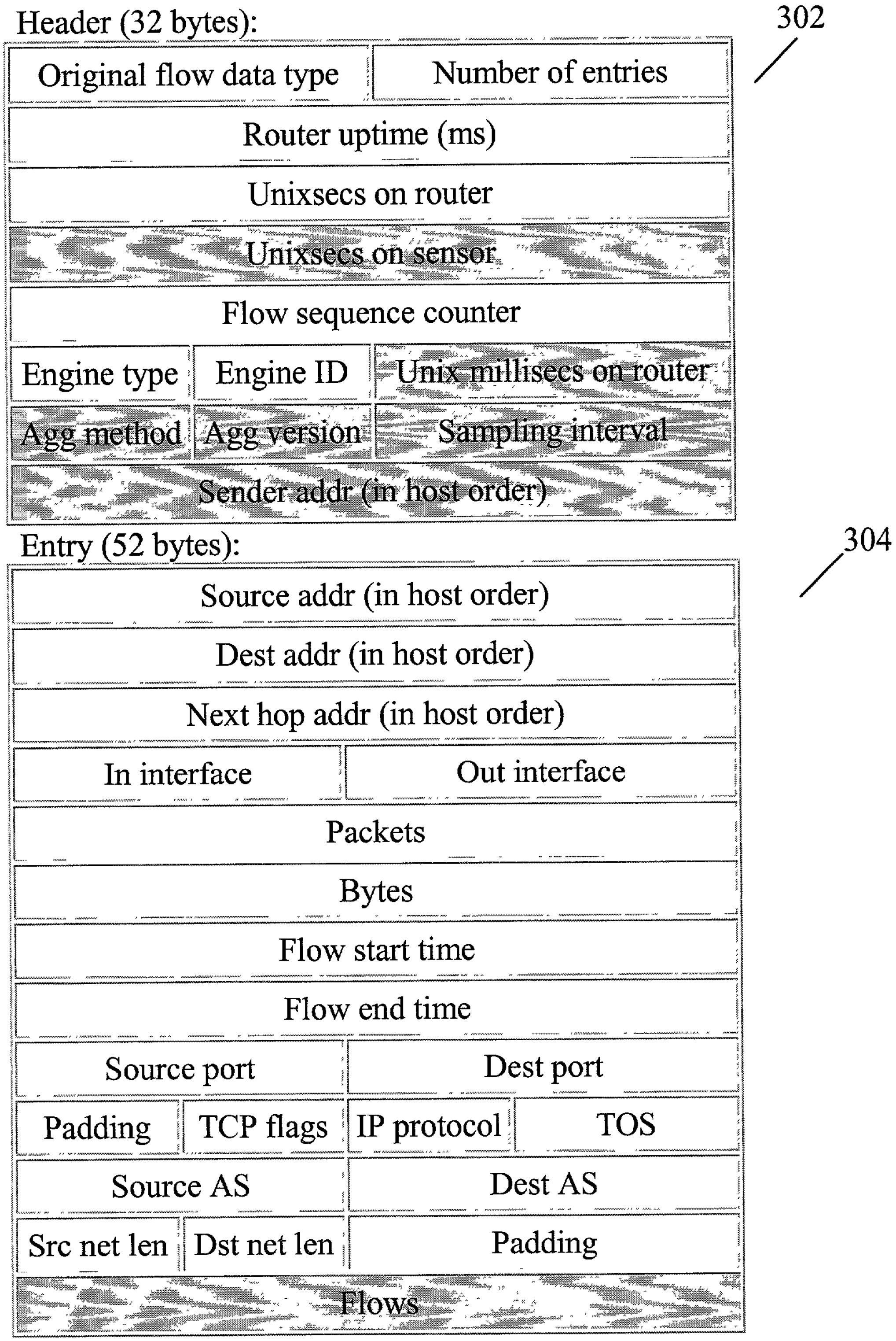

202
Receive Network Traffic Data
200
204
Determine Data Format
206
Analyze Timing Data
208
Timing Data Consistent?
210
No
Adjust Timing Data
Yes
212
Save Traffic Data [Converting Data Format]
214
Update Adjustment Data

Figure 2



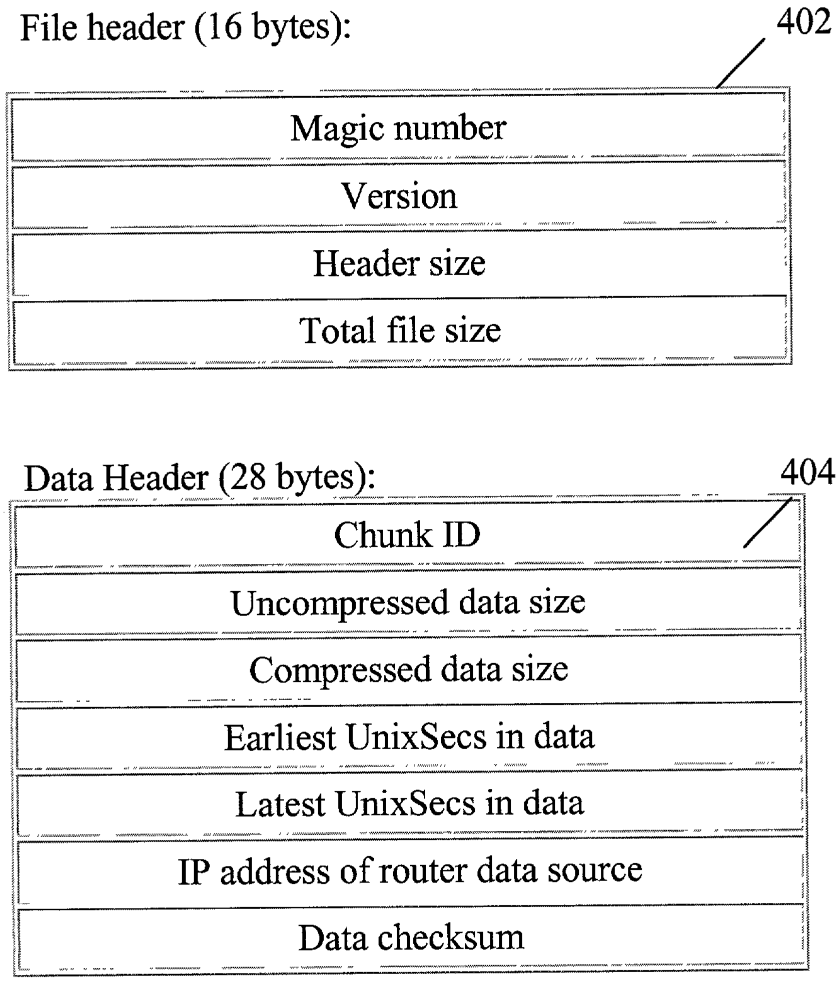
Header (32 bytes):

<table>
<tr><td colspan="2">Original flow data type</td><td colspan="2">Number of entries</td></tr>
<tr><td colspan="4">Router uptime (ms)</td></tr>
<tr><td colspan="4">Unixsecs on router</td></tr>
<tr><td colspan="4">Unixsecs on sensor</td></tr>
<tr><td colspan="4">Flow sequence counter</td></tr>
<tr><td>Engine type</td><td>Engine ID</td><td colspan="2">Unix millisecs on router</td></tr>
<tr><td>Agg method</td><td>Agg version</td><td colspan="2">Sampling interval</td></tr>
<tr><td colspan="4">Sender addr (in host order)</td></tr>
</table>

Entry (52 bytes):

<table>
<tr><td colspan="4">Source addr (in host order)</td></tr>
<tr><td colspan="4">Dest addr (in host order)</td></tr>
<tr><td colspan="4">Next hop addr (in host order)</td></tr>
<tr><td colspan="2">In interface</td><td colspan="2">Out interface</td></tr>
<tr><td colspan="4">Packets</td></tr>
<tr><td colspan="4">Bytes</td></tr>
<tr><td colspan="4">Flow start time</td></tr>
<tr><td colspan="4">Flow end time</td></tr>
<tr><td colspan="2">Source port</td><td colspan="2">Dest port</td></tr>
<tr><td>Padding</td><td>TCP flags</td><td>IP protocol</td><td>TOS</td></tr>
<tr><td colspan="2">Source AS</td><td colspan="2">Dest AS</td></tr>
<tr><td>Src net len</td><td>Dst net len</td><td colspan="2">Padding</td></tr>
<tr><td colspan="4">Flows</td></tr>
</table>

Figure 3



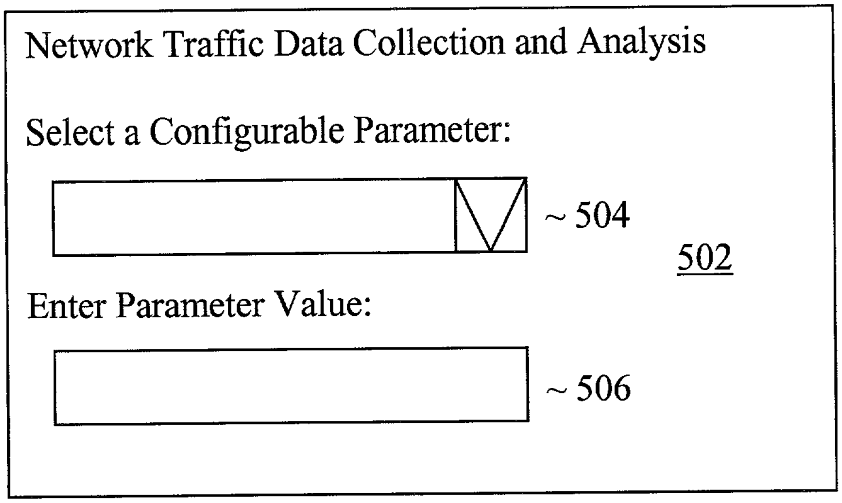
File header (16 bytes):

- Magic number
- Version
- Header size
- Total file size

Data Header (28 bytes):

- Chunk ID
- Uncompressed data size
- Compressed data size
- Earliest UnixSecs in data
- Latest UnixSecs in data
- IP address of router data source
- Data checksum

Figure 4

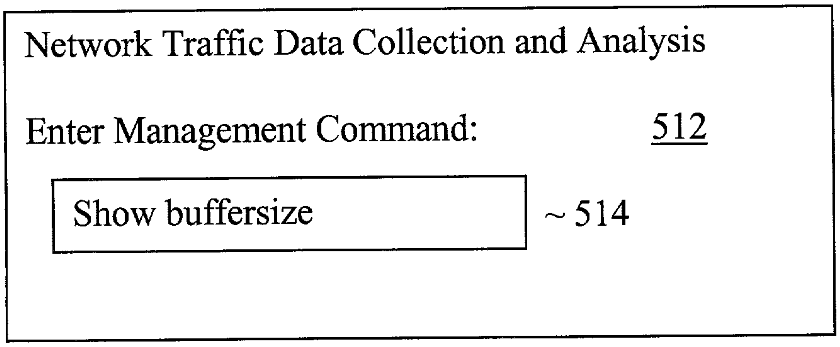


Network Traffic Data Collection and Analysis

Select a Configurable Parameter:

Enter Parameter Value:

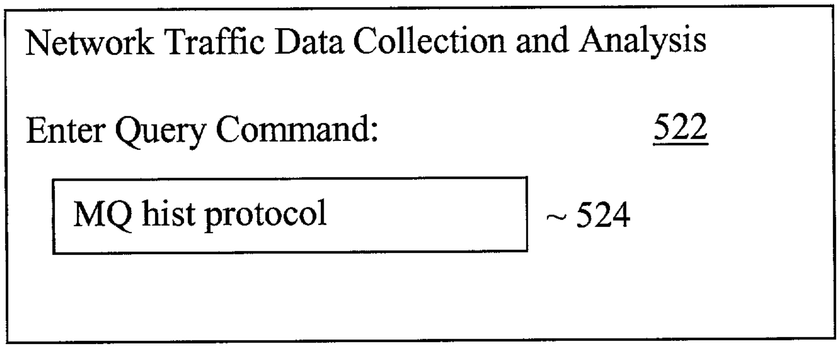
Figure 5a



Network Traffic Data Collection and Analysis

Enter Management Command: Show buffersize

Figure 5b

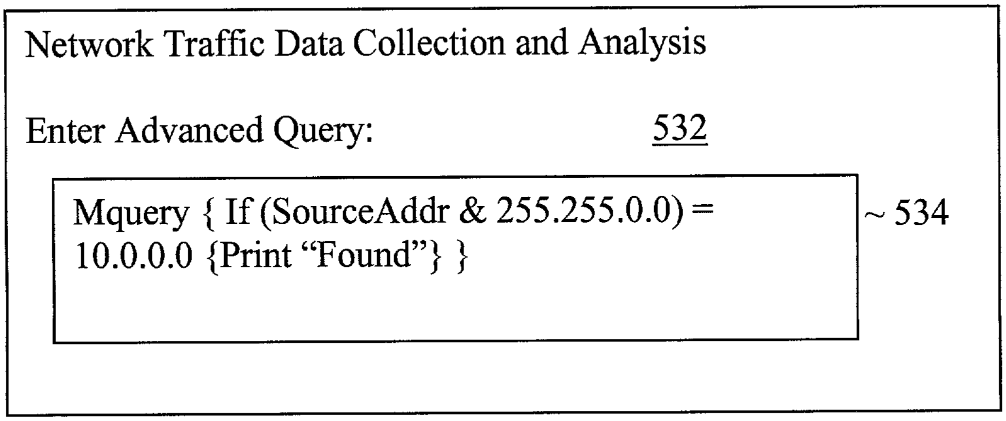


Network Traffic Data Collection and Analysis

Enter Query Command:

MQ hist protocol

Figure 5c



Network Traffic Data Collection and Analysis

Enter Advanced Query:

```
Mquery { If (SourceAddr & 255.255.0.0) =
10.0.0.0 {Print "Found"} }
```

Figure 5d

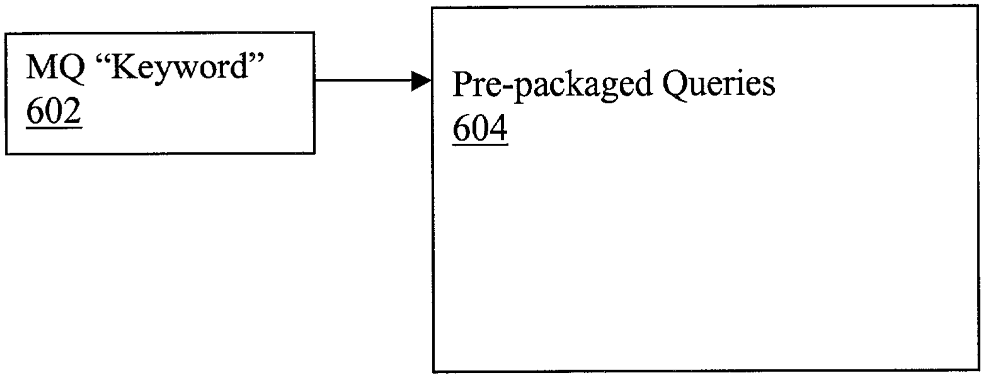


Figure 6a

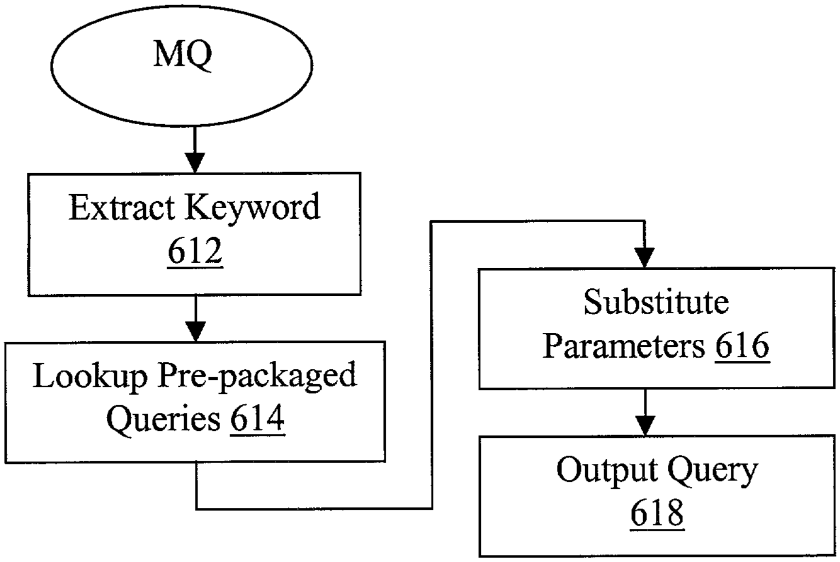


Figure 6b

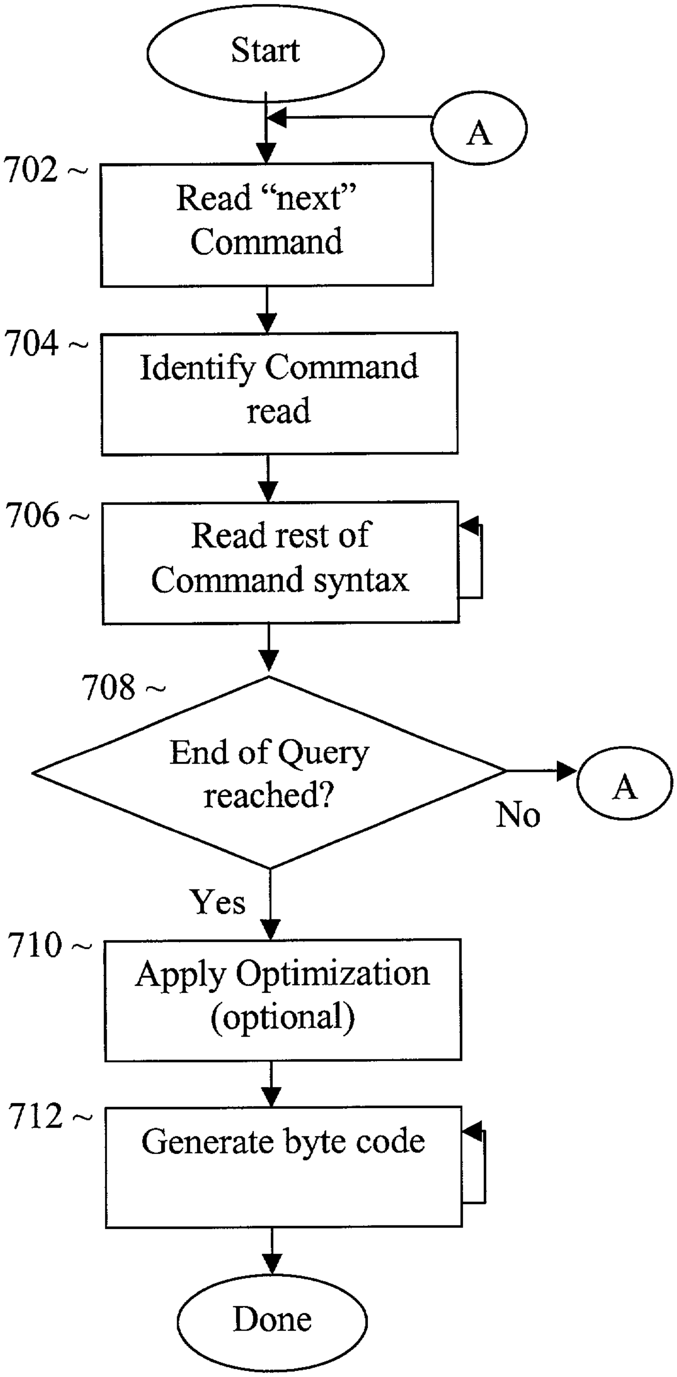

Start
A
702
Read "next" Command
704
Identify Command read
706
Read rest of Command syntax
708
End of Query reached?
No
A
Yes
710
Apply Optimization (optional)
712
Generate byte code
Done

Figure 7

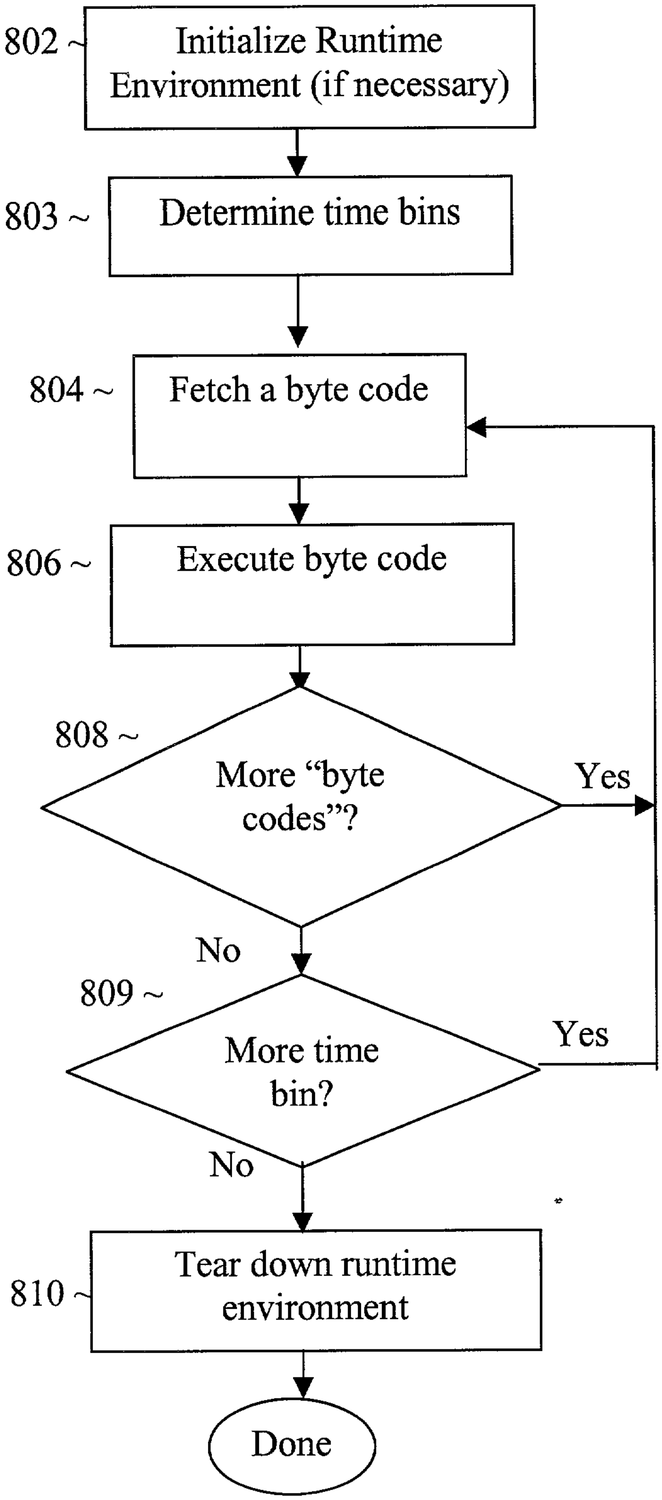

802 ~
Initialize Runtime Environment (if necessary)
803 ~
Determine time bins
804 ~
Fetch a byte code
806 ~
Execute byte code
808 ~
More "byte codes"?
Yes
No
809 ~
More time bin?
Yes
No
810 ~
Tear down runtime environment
Done

Figure 8

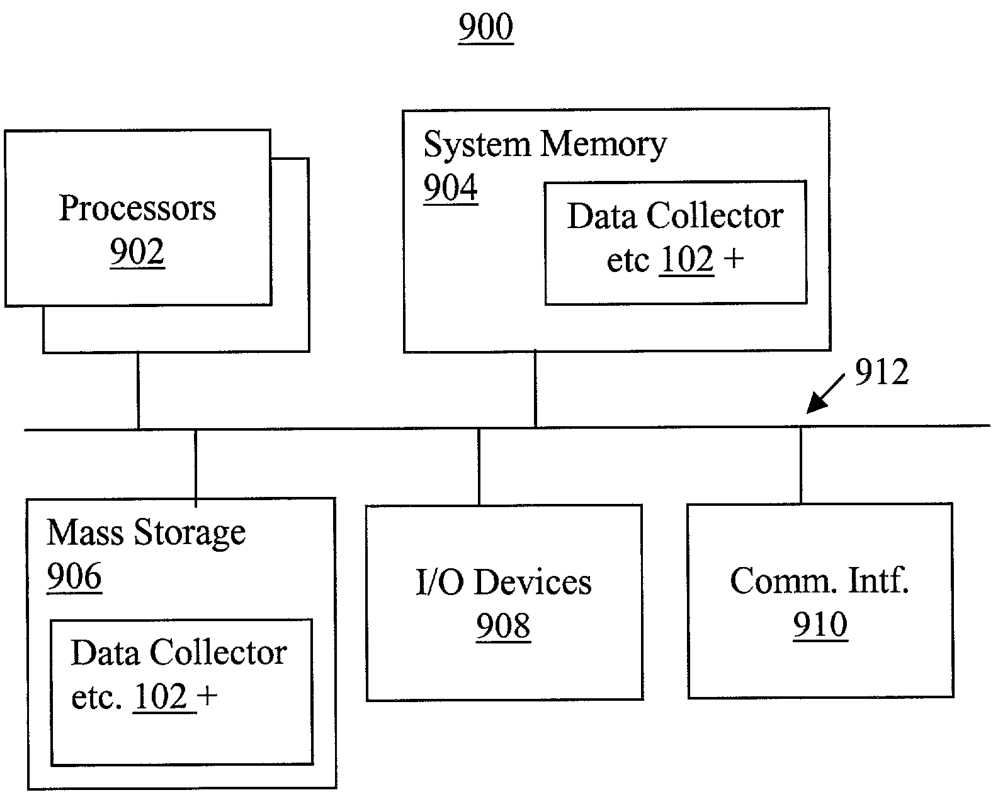

900
Processors
902
System Memory
904
Data Collector
etc 102 +
912
Mass Storage
906
Data Collector
etc. 102 +
I/O Devices
908
Comm. Intf.
910

Figure 9

```
// Copyright (c) 2000-2001 Asta Networks.  All rights reserved.

#ifndef __MARIO_QUERIES_HH__
#define __MARIO_QUERIES_HH__

enum QueryVersions
{
   MARIO_MAJOR_QUERY_VERSION    = 4,
   MARIO_MINOR_QUERY_VERSION    = 2,
   MARIO_QUERY_VERSION = ((MARIO_MAJOR_QUERY_VERSION
<< 4) + MARIO_MINOR_QUERY_VERSION)
};

enum Commands
{
   CMD_PRINT_SYSTEMVALUE   = 1,
   CMD_PRINT_NUMBER        = 2,
   CMD_PRINT_STRING        = 3,
   CMD_PRINT_NEWLINE       = 4,
   CMD_PRINT_HIST          = 5,
   CMD_PRINT_HIST_KEYS     = 6,
   CMD_SET_VAR             = 7,

   CMD_IF          = 8,
   CMD_IF_ELSE     = 9,

   WITH_FIRST_PACKET     = 10,
   WITH_LAST_PACKET      = 11,

   FOR_EACH_PACKET       = 12,
   FOR_EACH_FLOW         = 13,

   CMD_DEF_HIST          = 14,
   CMD_ADD_TO_HIST       = 15,
```

Figure 10a

```
  CMD_INCR_VAR          = 17,
  CMD_INCR_VAR_BY       = 18,

  CMD_INCR_LVAR         = 19,
  CMD_INCR_LVAR_BY      = 20,
  CMD_PRINT_LVAR        = 21,

  CMD_DEF_ARRAY                 = 22,
  CMD_ADD_TO_ARRAY              = 23,
  CMD_PRINT_ARRAY               = 24,
  CMD_PRINT_ARRAY_BY_PKT  = 25,
  CMD_PRINT_ARRAY_BY_FLOW= 26
};

enum NumericValues
{
  CONSTANT_BYTE_VALUE = 0x80,
  CONSTANT_INT_VALUE  = 0x81,
  HEADER_VALUE     = 0x82,
  FLOW_VALUE       = 0x83,
  SYSTEM_VALUE     = 0x84,
  VAR_VALUE        = 0x85,
  TCPFLAGS_VALUE = 0x86
};

enum HeaderValues
{
  HV_ORIGTYPE    = 0, // Original flow data type
  HV_COUNT       = 1, // The number of records
  HV_ROUTERUPTIME = 2, // Time in millisecs since router booted
  HV_ROUTERSECS  = 3, // Seconds since 0000 UTC 1970 on router
  HV_SENSORSECS  = 4, // Seconds since 0000 UTC 1970 on sensor
  HV_SEQNUM      = 5, // Seq counter of total flows seen
  HV_ENGINETYPE  = 6, // Type of interface generating the flows
  HV_ENGINEID    = 7, // ID of interface generating the flows
  HV_ROUTERMSECS  = 8, // Unix millisecs on router
  HV_AGGMETHOD   = 9, // Aggregation method (for NetFlow v8+)
```

Figure 10b

```
  HV_AGGVERSION   = 10, // Aggregation version (for NetFlow v8+)
  HV_SAMPINTERVAL = 11, // Sampling interval
  HV_SENDERADDR   = 12  // IP address where this data came from
};

enum FlowValues
{
  FV_SRCADDR  = 0,    // IP address of source
  FV_DSTADDR  = 1,    // IP address of destination
  FV_NEXTHOP  = 2,    // IP address of next-hop router
  FV_IN_IF    = 3,   // ID of incoming interface
  FV_OUT_IF   = 4,   // ID of outgoing interface
  FV_NUMPKTS  = 5,    // Number of packets in the flow
  FV_NUMBYTES = 6,    // Number of bytes in the flow
  FV_FIRST    = 7,    // On routerUptime scale, when flow started
  FV_LAST     = 8,    // On routerUptime scale, when flow ended
  FV_SRCPORT  = 9,    // Layer 4 source port
  FV_DSTPORT  = 10,   // Layer 4 destination port
  FV_PAD8     = 11,  // UNUSED
  FV_TCPFLAGS = 12,   // Or of all flags seen in flow, or ACK
  FV_PROTOCOL = 13,   // Layer 3 protocol
  FV_TOS      = 14,  // Type of service
  FV_SRC_AS   = 15,  // Source autonomous system
  FV_DST_AS   = 16,  // Destination autonomous system
  FV_SRC_MASK = 17,   // Number of valid src addr bits for netmask
  FV_DST_MASK = 18,   // Number of valid dst addr bits for netmask
  FV_PAD16    = 19,  // UNUSED
  FV_FLOWS    = 20   // Number of flows (when aggregated)
};

enum Operators
{
  OP_LGC_NOT        = 0xc0,
  OP_LGC_AND        = 0xc1,
  OP_LGC_OR         = 0xc2,
```

Figure 10c

```
  OP_BIT_NOT          = 0xc3,
  OP_BIT_AND          = 0xc4,
  OP_BIT_OR           = 0xc5,
  OP_BIT_XOR          = 0xc6,

  OP_EQ          = 0xc7,
  OP_NE          = 0xc8,
  OP_GT          = 0xc9,
  OP_GE          = 0xca,
  OP_LT          = 0xcb,
  OP_LE          = 0xcc,

  OP_ADD         = 0xcd,
  OP_SUB         = 0xce,
  OP_MUL         = 0xcf,
  OP_DIV         = 0xd0,
  OP_MOD         = 0xd1,
  OP_TRN         = 0xd2,

  OP_LVAR_MUL_DIV = 0xd3,

  OP_MUL_DIV_32 = 0xd4
};

enum PrintTypes
{
  PT_UINT        = 0,
  PT_INT         = 1,
  PT_IPADDR      = 2,
  PT_8BITS       = 3,
  PT_HEX         = 4,
  PT_PROTOCOL= 5,
  PT_TCPFLAGS= 6,
  PT_TM_MSECS= 7,
  PT_TM_SECS   = 8,
```

Figure 10d

```
  PT_CUINT      = 9,
  PT_CINT       = 10,
  PT_HEXBYTE = 11,
  PT_HEXWORD= 12,
  PT_BOOL       = 13,
  PT_SAMPINT  = 14,
  PT_HEXDWORD = PT_HEX
};

enum HistogramValueTypes
{
  HIST_SUM       = 0x71,
  HIST_OR        = 0x72,
  HIST_MAX       = 0x73,
  HIST_MIN       = 0x74,
  HIST_FIRST     = 0x75,
  HIST_LAST      = 0x76,
  HIST_UNIQUE = 0x77
};

enum SystemValues
{
  SYSVAL_VERSION_STRING     = 0,
  SYSVAL_CURRENT_TIME        = 1,
  SYSVAL_DATA_PRESENT        = 2
};

#endif // __MARIO_QUERIES_HH__
```

Figure 10e